Preprint
UCRL-JC-137053

KULL: LLNL's ASCI Inertial Confinement Fusion Simulation Code

J.A. Rathkopf, D.S. Miller, J.M. Owen, L.M. Stuart, M.R. Zika, P.G. Eltgroth, N.K. Madsen, K.P. McCandless, P.F. Nowak, M.K. Nemanic, N.A. Gentile, N.D. Keen and T.S. Palmer

This article was submitted to
Physor 2000 American Nuclear Society Topical Meeting on Advances in Reactor Physics and Mathematics and Computation into the Next Millennium
Pittsburgh, PA
May 7-11, 2000

U.S. Department of Energy

Lawrence
Livermore
National
Laboratory

January 10, 2000

KULL: LLNL'S ASCI INERTIAL CONFINEMENT FUSION SIMULATION CODE

James A. Rathkopf, Douglas S. Miller, John M. Owen, Linda M. Stuart, Michael R. Zika,
Peter G. Eltgroth, Niel K. Madsen, Kathleen P. McCandless, Paul F. Nowak,
Michael K. Nemanic, Nicholas A. Gentile, and Noel D. Keen
Lawrence Livermore National Laboratory
P.O. Box 808, L-18
Livermore, California 94551-0808
rathkopf@llnl.gov; dougmiller@llnl.gov; mikeowen@llnl.gov; lstuart@llnl.gov; zika@llnl.gov;
peltgroth@llnl.gov; nkmadsen@llnl.gov; mccandless2@llnl.gov; pnowak@llnl.gov;
nemanic@llnl.gov; gentile@llnl.gov; keen3@llnl.gov

Todd S. Palmer
Department of Nuclear Engineering
Oregon State University
Corvallis, Oregon 97331
palmerts@fermi.ne.orst.edu

ABSTRACT

KULL is a three dimensional, time dependent radiation hydrodynamics simulation code under development at Lawrence Livermore National Laboratory. A part of the U.S. Department of Energy's Accelerated Strategic Computing Initiative (ASCI), KULL's purpose is to simulate the physical processes in Inertial Confinement Fusion (ICF) targets. The National Ignition Facility, where ICF experiments will be conducted, and ASCI are part of the experimental and computational components of DOE's Stockpile Stewardship Program. This paper provides an overview of ASCI and describes KULL, its hydrodynamic simulation capability and its three methods of simulating radiative transfer. Particular emphasis is given to the parallelization techniques essential to obtain the performance required of the Stockpile Stewardship Program and to exploit the massively parallel processor machines that ASCI is procuring.

1. INTRODUCTION

With the end of underground nuclear testing, the United States must rely solely on non-nuclear experiments and numerical simulations, together with archival underground nuclear test data, to certify the continued safety, performance, and reliability of the nation's nuclear stockpile. The U.S. Department of Energy (DOE) and its three Defense Programs nuclear weapons national laboratories–Los Alamos, Lawrence Livermore, and Sandia–have established the science-based Stockpile Stewardship Program to meet this challenge. Large experimental facilities, such as the

National Ignition Facility (NIF) currently under construction at Lawrence Livermore, allow scientists to study physics in physical regimes approximating the conditions inside a nuclear explosion. The Accelerated Strategic Computing Initiative, or ASCI, provides virtual testing and prototyping capabilities based on advanced simulation codes and high-performance computing.

When completed, the football-field-sized NIF (http://www-lasers.llnl.gov/lasers/nif.html) will focus 192 laser beams on a mm-sized target ablating the outer surface of a tiny pellet containing deuterium and tritium. The ablation compresses the pellet which results in large increases in pressure, density, and temperature, conditions similar to those in an exploding nuclear weapon or the interior of a star. With sufficiently high temperatures, the deuterium and tritium nuclei can overcome their coulomb repulsion causing a fusion reaction that releases 17.6 MeV. Ignition, the technical goal of NIF and the Inertial Confinement Fusion (ICF) community, is achieved when the heating resulting from fusion is sufficient to sustain the continuing chain reaction.

The goal of ASCI is to provide both the software, in the form of the operating environment and the simulation codes, and the hardware, in the form of massively parallel processor supercomputers, necessary for predicting the processes occurring in ICF targets and nuclear weapons.

The hardware component of ASCI represents a partnership between DOE and the computing industry. The first supercomputers were developed in the 1960s for weapon applications as part of such a partnership. This relationship continued for several decades until the late 1980s and early 1990s when the weapons laboratories influence on high performance computing development waned as their needs decreased. Meanwhile, the computer industry shifted their focus to the business market where raw performance, especially on scientific applications, was less important than the price-performance ratio. With stockpile stewardship's reliance now on simulation, ASCI can encourage the development of the highest performing platforms by both partnering in development as well as by providing a highly visible market for the machines once developed. ASCI *accelerates* a computer company's normal development cycle by demanding performance beyond the requirements of conventional customers. The company benefits by developing market products that perform above that which would have been available if it were not for ASCI-partnered development. This development strategy is possible because ASCI partners use commodity, off-the-shelf hardware, specifically micro-processors with unique interconnects, rather than developing completely specialized, one-of-a-kind machines.

Massively parallel processor architectures are the only way to achieve the performance demanded by ASCI physics simulation codes. To date, three such systems have been delivered, one to each of the Defense Programs weapons laboratories. Intel delivered the first, ASCI Red (http://www.sandia.gov/ASCI/Red/main.html) to Sandia National Laboratory in Albuquerque, NM, in late 1996. The full machine, completed in June 1997 (see Figure 1), consists of 4640 dual Pentium Pro processor compute nodes with nearly 600 Gbytes of memory and an aggregate peak performance of 1.3 TeraOps (10^{12} floating point operations per second). In early 1997, Silicon Graphics and IBM made the initial deliveries of the next generation ASCI machines to Los Alamos and Lawrence Livermore, respectively. By late 1998, the final configurations were in place and delivered aggregate peak performances of over 3 TeraOps each and measured sustained performance of over 1 TeraOps. Los Alamos's Blue Mountain machine (http://www.lanl.gov/asci/bluemtn/bluemtn.html and http://lib-www.lanl.gov/la-pubs/00418752.pdf) consists of 48 Silicon Graphics Origin 2000 shared memory multi-processor nodes (see Figure 2). Each node has 128 250-MHz processors for a total of 6144 processors and 1.5 Terabytes of memory. Lawrence Livermore's Blue Pacific machine (http://www.llnl.gov/asci/platforms/bluepac) consists of 1464 shared memory multi-processors, each node with four 332-MHz IBM PowerPC 604e processors (see Figure 3). The machine has a total of 5856 processors and 2.5 Terabytes of memory.

The next step toward the ASCI's eventual goal of a 100 TeraOps system by 2004 is the Option White machine being manufactured by IBM and to be delivered in stages to Lawrence Livermore throughout 2000. Option White will have of a peak aggregate performance of 10 TeraOps over 512 shared memory nodes, each consisting of 16 processors.

The simulation codes that make up the bulk of the software component of ASCI include classified nuclear performance codes that can predict the details of an exploding nuclear weapon, unclassified materials modeling codes that employ molecular dynamics to study the long-term degradation of material under the influence of low-level radiation, and unclassified multi-physics codes that simulate the processes within an ICF pellet. KULL, the subject of this paper, is one such ICF modeling code used by scientists to design experiments for NIF and to study physics in regimes similar to that seen in nuclear explosions.

2. KULL OVERVIEW

KULL, a simulation code that has been under development by an interdisciplinary team of scientists, engineers and computer scientists at Lawrence Livermore for the last several years, can model the time-dependent processes occurring in an ICF pellet. It must model the fluid motion within the pellet as it undergoes tremendous increases in temperature, pressure, and density; it must model the transport of x-ray radiation emitted by high-temperature material; it must model the release of energy and depletion of reactants resulting from the fusion process.

2.1. PHYSICS SIMULATION

Fluid motion is simulated through the numerical solution of the compressible hydrodynamics equations. The solution algorithm must be able to accurately model the effect of large density gradients, shocks, and various kinds of instabilities. KULL uses the Arbitrary Lagrange-Eulerian (ALE) approach for this component. Section 3 describes KULL's hydrodynamics package in more detail.

Hot matter radiates x-rays that propagate throughout the system causing a redistribution of energy as the photons are absorbed and heat up other regions. These regions, in turn, emit more x-rays. The absorption cross section or opacity of material is strongly dependent on its composition and temperature, making x-ray radiation transport a highly non-linear process. Users of KULL may choose one of three methods of modeling radiation flow: flux-limited diffusion, deterministic discrete ordinate transport, and a specialized Monte Carlo method known as Implicit Monte Carlo. Details of the three methods are described in subsequent sections.

KULL can also simulate the fusion process. Given a composition, density, and temperature, KULL's thermonuclear burn package can calculate the reaction rate density, rate of energy release, the consumption of reactants, and production of reaction products.

In addition to its suite of physics simulation packages, KULL has several unique features compared to other, earlier ICF simulation codes. Most importantly, KULL models an experiment in full three dimensions, where earlier codes exploit approximate axial symmetry and model only two dimensions. KULL's mesh or zoning scheme also differs from most other codes of this type. Instead of an ordered arrangement of a few types of polyhedra, typically degenerate hexahedra (the so-called finite element zoo), KULL supports an unstructured mesh of fully arbitrarily connected zones of fully arbitrary polyhedra. While this severely complicates the numerical solution algorithms, this generality allows nearly infinite flexibility in a user's choice of zoning, enabling more accurate geometric representation of experiments. It also helps transfer the burden of zoning a problem from the user onto the mesh generation software. Figure 4 shows an example of such a grid.

2.2. COMPUTER SCIENCE FEATURES

KULL's choice of the object-oriented C++ programming language also represents an advance in the state of the art of physics simulation code development. KULL makes heavy use of the language's templating ability to allow for efficient polymorphism, one of the advantages of object oriented design. Python, an object-oriented scripting language, allows KULL developers to rapidly prototype software changes and provides a means for users to "steer" their simulations as they develop. Although the core infrastructure of KULL and many of its physics packages are written in C++, some package authors have chosen Fortran90 instead.

As mentioned above, only by exploiting massively parallel computer architectures can an ASCI code hope to meet the demands of predictive simulation. Some of these simulations may require hundreds of millions of zones. KULL's algorithms have been designed to run efficiently on these platforms. KULL uses a strategy of mixed parallelization techniques. On ASCI Blue Pacific, KULL's primary target machine, message passing through the Message Passing Interface (MPI)[1] is used to communicate among the four-processor SMP compute nodes. Within an SMP node either MPI or, for more efficient parallel performance, threading through OpenMP[2] directives provide parallelization.

3. HYDRODYNAMICS

Traditionally, computational hydrodynamics methods have fallen into two camps. The first, known as Eulerian methods, hold the mesh of zones stationary and fluid flows from one zone to another through zone faces via advection. Eulerian methods are extremely robust, capable of running under severe flow conditions. These methods are usually implemented on a regular, structured mesh that permits very efficient solution schemes. Because the zones do not move, zone tangling in flows of high vorticity, a problem with other schemes, is impossible by definition. Eulerian methods can be subject to inaccuracies due to numerical diffusion as interfaces between materials flow across zones unless sophisticated interface tracking techniques are employed. Finally, if no adaptive techniques are used, the zone resolution specified at the beginning of a simulation is constant over time. This may result in poor resolution of details in a converging geometry such as an imploding ICF pellet.

At the other end of the scale are Lagrangian methods. Here, zones flow with the fluid and no fluid crosses zone faces. Interfaces remain intact as they travel with zones. Again, because the zones travel with the fluid, highly resolved regions will retain their high resolution throughout the flow evolution. Unfortunately, in flow fields with high vorticity, the mesh may tangle resulting in zones that are contorted to such a degree that their volumes are negative as they are turned inside out.

Over the last decade or so, another method has emerged that smoothly spans these two extremes offering the benefits of both: the Arbitrary Lagrange-Eulerian (ALE) method. In ALE, the solution algorithm can vary from pure Eulerian to pure Lagrangian through the adjustment of a parameter that may be set by the user or set automatically according to some established criteria that may vary throughout the geometry of the simulation. An ALE single solution step consists of two parts. First, in the Lagrange step, the mesh travels with the fluid as determined by the equations of momentum and energy conservation and the length of the time step. Then, in the Eulerian remap step, the mesh is moved some fraction of the way back to its original position. As the mesh moves through the fluid, fluid advects through the faces of the mesh. The fraction is determined by the adjustable parameter noted above. The mesh tangling that plagues simple Lagrangian schemes can be avoided by advecting more in regimes of high vorticity. The difficulty of tracking interfaces seen in simple Eulerian schemes can be eliminated by avoiding the advection step near material interfaces. In practice, however, ALE schemes employ interface tracking techniques to allow the interface to traverse the mesh without undue numerical dispersion.

KULL currently implements both Lagrangian and ALE hydrodynamics, with plans in the future of implementing more exotic techniques such as Free Lagrange. Due to the complexities of the fluid flow encountered in ICF calculations, techniques must be robust to turbulent flows. Figure 5 shows such a complex flow simulated with KULL’s ALE. Both the Lagrange and ALE remap steps are second-order accurate in space and time, where the Lagrange cycle solves the standard conservation equations for mass, momentum and energy:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\rho \nabla \bullet \mathbf{v} \\
\frac{\partial \rho v_k}{\partial t} &= -\frac{\partial P}{\partial x_k} - \rho v_k \nabla \bullet \mathbf{v} \\
\frac{\partial \rho \varepsilon}{\partial t} &= -P \nabla \bullet \mathbf{v} - \rho \varepsilon \nabla \bullet \mathbf{v}
\end{aligned}$$

KULL’s ALE mode consists of an arbitrary number of Lagrange steps followed by an advective remap step to relax the mesh. KULL allows a number of choices for how the mesh should be moved during the remap phase, ranging from nearly Lagrangian (where the mesh closely follows the fluid) to purely Eulerian (where the mesh is remapped to its original configuration after every Lagrange step). The user is free to mix different mesh motion algorithms in different regions of a problem. The ALE remap for any quantity q (which could be the mass, momentum, or energy or any other quantity) can be expressed as a differential over a pseudo-time τ,

$$\frac{\partial q}{\partial t} = \mathbf{v}_g \bullet \nabla q,$$

where $\mathbf{v}_g$ represents the motion of the mesh (grid).

The most novel aspect of the current hydrodynamic schemes in KULL is that they are designed to work well on unstructured meshes. As an example of this capability, Figure 6 shows an ICF implosion simulation run on a mesh generated using a Voronoi cell structure (the dual mesh of a Delauny triangulated mesh).

4. RADIATION TRANSPORT

The equations for thermal radiative transfer describe the transport, absorption, scattering, and emission of x-rays within a physical material. As photons propagate, they are both absorbed and emitted by the material altering the temperature of the material, which in turn changes the properties of the material. These changes in the material properties subsequently alter the rate and the energy spectrum of emission, as well as the photon mean-free-path. These physical processes are modeled with the Boltzmann transport equation for photons and an energy balance equation for the material. The photon transport equation is reminiscent of the neutron transport equation but with one significant difference: the emission source on the right side is proportional to the absorption cross section and the Planck function,

$$\left(\frac{1}{c}\frac{\partial}{\partial t} + \Omega \cdot \nabla + \sigma\left(\mathbf{r}, E, T(t), \rho(t)\right)\right) I(\mathbf{r}, \Omega, E, t) = \sigma\left(\mathbf{r}, E, T(t), \rho(t)\right) B\left(\mathbf{r}, E, T(t)\right).$$

Here, for simplicity there is no scattering and no external source. The Planck function is written

$$B(\nu, T) = \frac{2h\nu^3}{c^2}\left(e^{\frac{h\nu}{kT}} - 1\right)^{-1}$$

where

$$E = h\nu .$$

The material energy balance equation shows that the rate of change in material energy density, u_m, is the difference of the rate of energy absorbed from the radiation field and the rate of energy emitted by the material back into the radiation field:

$$\frac{\partial u_m\left(\mathbf{r}, T(t), \rho(t)\right)}{\partial t} = \int_0^\infty dE \sigma\left(\mathbf{r}, E, T(t), \rho(t)\right)\left[\int_{4\pi} d\Omega I(\mathbf{r}, \Omega, E, t) - B\left(\mathbf{r}, E, T(t)\right)\right].$$

The fundamental unknowns are the radiation intensity I and the material temperature T in a seven dimensional phase space (three in position $\mathbf{r}$, two in direction $\boldsymbol{\Omega}$, one in energy E, and one in time t). These equations exhibit strong nonlinear coupling through the emission terms and the material opacity or cross section. The material opacity itself is a very strong function of temperature, photon energy, and atomic number and can range over ten orders of magnitude in a single calculation. Fine resolution in space, energy and direction is required to adequately represent the photon distribution in time. A thorough discussion of the above equations and the physics they represent may be found in reference 3.

5. DIFFUSION RADIATION TRANSFER

Both the unstructured grid capability of KULL and the strong coupling between the matter temperature and the radiation field pose unique challenges to simulating these processes. A method based on the flux-limited, multi-group diffusion approximation to the angular-dependent transport equation provides users with the capability of performing more efficient, albeit less accurate, simulations than possible with full-fledged transport methods.

A point centered diffusion differencing scheme[4] has been developed for three dimensional unstructured meshes that has the following attractive properties: 1) equivalence with the standard seven-point point centered scheme on an orthogonal mesh; 2) preservation of the homogeneous linear solution; 3) second-order accuracy; 4) strict conservation within the control volume surrounding each point; and 5) convergence to the exact result as the mesh is refined, regardless of the smoothness of the mesh. A potential disadvantage of the method is that the diffusion matrix is asymmetric, in general.

The scheme is based on dividing each polyhedral zone into subzone volumes called *wedges*. Each vertex has twice as many wedges as it has edges emanating from it. Six wedges are associated with each corner of a hexahedral zone. A wedge is a tetrahedron with vertices at a zone center z, the point defining the corner p, a face center f, and an edge center e as shown in Figure 8. Burton[5] originally suggested the concept of tetrahedral subzone volumes in his hydrodynamics work. Particle balance is enforced by integrating the diffusion equation,

$$\nabla \bullet \mathbf{J} + \sigma\phi = Q,$$

over the control volume formed by the union of wedges surrounding a point. Here, Q is an arbitrary source, the intensity integrated over all angle is the flux,

$$\phi\left(\mathbf{r}, E, t\right) = \int_{4\pi} d\Omega I(\mathbf{r}, \Omega, E, t),$$

and the first moment of the intensity is the current,

$$\mathbf{J}\left(\mathbf{r}, E, t\right) = \int_{4\pi} d\Omega\Omega I(\mathbf{r}, \Omega, E, t).$$

Area-averaged currents are eliminated in favor of fluxes through Fick’s law,

$$\mathbf{J} = -D\nabla\phi .$$

The resulting gradients of flux can be expressed in terms of fluxes at the wedge vertices. Then, by imposing the requirement of continuity of the normal component of current across wedge faces and defining zone-center fluxes as the inverse-length-weighted average of extrapolations of point fluxes, a matrix results where each point is connected to every point associated with the zones surrounding that point. The elimination of zone-, face- and edge-centered fluxes in favor of point-centered fluxes on volume elements that are not right hexehedra can result in the diffusion matrix being asymmentric.

5.1. PARALLEL DIFFUSION MATRIX SOLUTION STRATEGY

The possibility of an asymmetric matrix eliminates from consideration the use of some conventional matrix solvers. KULL's diffusion package has implemented in it two standard linear Krylov subspace (iterative) solver libraries for matrix solution: PETSc[6]and HYPRE[7]. Because the underlying matrix is asymmetric, KULL uses the GMRES (Generalized Minimum Residual) method that accommodates asymmetric matrices. An excellent discussion of this and other Krylov subspace methods may be found in reference 8. To accelerate convergence of the matrix solution the linear solver libraries provide several options for preconditioning. We have found that the algebraic multigrid[9] (AMG) preconditioner available in HYPRE is robust for a wide range of problems and scales well to many (thousands) of processors. The block-Jacobi preconditioner available in PETSc also works well for a large class of problems; however, it does not scale as well as AMG.

PETSc and HYPRE utilize spatial domain decomposition to exploit parallel processor hardware. Each domain is handled by separate processes. The solution of the diffusion matrix is found by iterating through a subspace of limited dimensionality. This involves communication at surfaces of domains. The matrix equation is transformed in a preconditioning step, essentially local to the subdomain, in order to obtain good convergence properties for the global solution. Convergence testing and iteration control are performed using simple and efficient global reduction operations.

5.2. COUPLED EQUATION SOLUTION METHOD

The solution of the diffusion matrix equation represents only one part of the total solution process. The coupling of the material energy balance equation with the diffusion equation through the emission and absorption terms must be considered. A brief discussion of the solution procedure follows. The scheme employs a fully implicit time discretization with the exception that the opacities and specific heats are evaluated at the beginning-of-timestep temperatures. To obtain an implicit Planckian, we use a Newton-Raphson iteration (with iteration index *k*) based on the following first order expansion:

$$B(T_{k+1}) \approx B(T_k) + (T_{k+1} - T_k)\left.\frac{\partial B}{\partial T}\right|_{T_k} .$$

Substitution of this into the discretized form of the transport and energy balance equations leads to a linear steady-state multigroup problem to be solved for each Newton-Raphson iteration on each time step. The flowchart in Figure 7 displays the overall iteration strategy for each time step. Starting with a beginning of time step temperature and material properties, we evaluate the emission source and then solve for the intensity in each energy group over the entire spatial domain. This is followed by an update of the material temperature and finally a convergence test on the temperature and the integrated intensity. If convergence is satisfied, we advance to the next time step. Otherwise, we continue the temperature iteration loop.

6. DISCRETE ORDINATE RADIATION TRANSPORT

The discrete ordinate, or S_N, method of solving the full angular-dependent radiation transport equation experiences the same challenges as the diffusion approximation, namely an unstructured mesh and strong matter-radiation coupling, with the additional calculational complexity of resolving the angular dimensions of phase space.

The discretization scheme used in KULL's S_N module TETON is upstream corner-balance in space, multigroup in energy, discrete ordinate in angle, and fully implicit in time. The upstream corner-balance scheme employs a means of dividing the polyhedral zones into subvolume cells similar to that used by the diffusion method described in the previous section. Details of the spatial discretization have been given elsewhere[10] and will not be described here. The system is solved using the grey transport acceleration method (GTA)[11] except that an S_2 quadrature is always used for acceleration. The resulting one group and S_2 problem is solved using transport synthetic acceleration (TSA).[12]

The coupled system of matter energy and radiation equations is solved in a manner similar to the one used for diffusion. But, because a packaged parallel matrix solver is inappropriate for the transport solution, parallelization is explicitly considered in the construction of the transport solver. On the highest level, the spatial mesh is divided into multiple sub-domains, one for each compute node on ASCI Blue Pacific. With the exception of the means of solving for intensities, the spatial domain decomposition and the transport iteration strategy[13] shown in Figure 9, is the same as that for diffusion. After finding the intensities in each direction and energy group on a node's spatial domain, the boundary intensities are communicated between nodes. A node sends its exiting intensities to and receives incoming intensities from nodes containing neighboring domains. This is followed by a "local" acceleration step where there is no internode communication. Testing for global convergence of temperatures and radiation intensities requires internode communication.

6.1. PARALLEL SOLUTION STRATEGY

While it is possible to distribute the six dimensional calculation over compute nodes via decomposition in a variety of combinations of space, angle, and energy, spatial domain decomposition best fulfils the following criteria:

- It must optimize both the high-order problem (multigroup, S_N) and the low-order problem (one group, S_2) used for iterative acceleration.
- It must scale to problems with billions of unknowns using thousands of processors.
- It must handle multigroup physics constraints (e.g., group to group scattering).

- It should try to preserve single processor convergence rates.
- The communication costs should be low and not increase substantially with the number of domains.
- It should minimize total computation time and storage.
- The resulting computation on a single node should be as cache efficient as possible.

Energy decomposition fails because the acceleration step cannot be decomposed and because it requires that each node hold the entire mesh, thus severely limiting the problem size. Decomposition in angle suffers these same limitations, and also requires frequent communication points.

On Blue Pacific, each compute node contains four SMP processors. To fully extract the power of such a machine, efficient use at the SMP level must also be pursued. Two distinct, but intimately connected aspects must be considered: using multiple threads and optimizing cache performance.

The criteria for developing a SMP algorithm using threads are similar to the ones chosen for deciding how to spatially distribute the problem. In the case of threads the primary concern is developing an algorithm that automatically handles both the high-order problem (multigroup, S_N) as well as the low-order acceleration step (one group, S_2). Threading over angles satisfies this constraint, while threading over blocks of energy groups does not.

Threading is accomplished using a single OpenMP directive (“parallel do”) on the loop over angles. This loop encompasses the source calculation, the sweep through the spatial grid and, as we will see below, the loop over energy groups. Both the solutions for the intensities and the acceleration calculation use this angle loop. The goal is to put as much work as possible in the threaded loop and our experience has shown that this is critically important. (An initial attempt at loop level directives provided no more than a speedup of 1.5 out of 4 processors.)

6.2. CACHE OPTIMIZATION

Due to the fact that unstructured meshes have arbitrary connectivity between grid elements, there is no simple relationship that maps from one subzone to its neighbors. In order to allow for completely arbitrary zone shapes and topologies, it is necessary to compute an ordered list of subzones for each discrete ordinate direction. As a result, indirect memory references are used throughout the lowest level of the algorithm, where the grid is “swept” for each direction. Indirect memory references are essentially random walks through memory and can result in a cache miss for every memory reference and almost no reuse of the data that resides in cache. Figure 10 shows the cumulative effect of several techniques used to improve this situation. In general, the improvements described below reduced the number of data cache misses and thus resulted in shorter run times by providing a larger fraction of “useful” cycles.

All runs were made on a DEC Alpha, gathering the CPU time, the percent of “useful” cycles and the number of cache misses for a benchmark problem run with stand-alone versions of KULL’s TETON package. The first data point on the left of each curve was obtained by running the vector version of TETON with source optimized for the Cray YMP. This serial benchmark ran in 3.7 minutes on a Cray YMP and 23.2 minutes on a DEC Alpha. After making the code modifications identified in Figure 10 and described below, the run time for the threaded code was reduced to 0.9 minutes (four CPUs) on the DEC Alpha and 1.5 minutes on an Blue Pacific computing node (four 604e CPUs).

The most dramatic improvement resulted from converting from a vector version of the code to a scalar version. Arrays were replaced with scalars in arithmetic operations; small vector loops were “fused” into larger single scalar loops.

Another large improvement was gained by storing the connectivity information such that one array reference can bring into cache all of the integer data that is required to calculate a single intensity (e.g., list of neighboring subzones, zone index, face index, etc.). The same grouping of data can be done for the real arrays (e.g., incident intensities and sources).

A third remedy is to “prefetch” the data that is required to calculate an intensity. Before entering the loop over subzones, TETON loads all of the data required to compute the first intensity. Once inside the loop, the intensity for subzone *i* is computed while the data required for the intensity at *i*+1 are loaded. In this fashion, stalls resulting from slow memory access are amortized over concurrent calculation. More dramatic gains can be achieved from maximizing the reuse of data that is already loaded, for example all of the geometry information. One way to do this is by collecting or “batching” all photon energy groups that have the same set of angles and transporting them together.

6.3. PARALLEL PERFORMANCE

Combining the two techniques (spatial decomposition over nodes via MPI message passing and threading over angle on the processors on each node using OpenMP) results in a scheme that maximizes the performance of Blue Pacific. It should be noted that the calculations still take a long time to complete due to the huge number of unknowns to solve. Without these algorithm developments and machines like Blue Pacific, calculations required by the Stockpile Stewardship Program would be impossible.

Figure 11 shows run time as a function of the number of processors (the number of subdomains is one-fourth the number of processors) for four fixed size problems. The problems range in size from a single group, S_4 calculation on $3.2x10^5$ zones (~$6.1x10^7$ unknowns) to an eight group, S_6 calculation on $1.024x10^6$ zones (~$3.1x10^9$ unknowns). All four runs show good scaling, having a parallel efficiency of about 90% for the largest processor count compared to the run time per CPU for the smallest processor count. (The problems require too much memory to be run on fewer processors than the minimums shown in Figure 11.)

Figure 12 compares constant workload per processor runs made with MPI processes alone and with the combination of MPI and threads (4 threads per node). The number of subdomains equals the number of processors with MPI alone and equals one-fourth the number of processors with MPI and threads. The fundamental size problem given to a single processor is an eight energy group and S_6 (48 angles) calculation on 2000 zones. As the total number of CPUs is increased, two effects are seen in the MPI only curves. The upper curve, representing 4 MPI processes per node, shows increased CPU time at 128 CPUs due to the “saturation” of the communication sub-system. There is a similar rise in the middle curve, representing 1 MPI process per node. At 256 CPUs, the top curve suffers from an additional cost due to the contention among the 4 MPI processes per node for the single memory bus. The effect of “stressed” communications and contention for memory seem to synergize in the upper curve at 256 CPUs. This is not apparent on the middle curve since it has only 1 MPI process accessing memory. The first effect is not present at all in the bottom curve, representing 1 MPI process with 4 threads, until the 512 CPUs are used, but even then the “strain” is only marginally significant.

7. IMPLICIT MONTE CARLO METHOD OF RADIATION TRANSPORT

KULL’s third option for determining the radiation field is through a type of Monte Carlo method known as Implicit Monte Carlo[14] (IMC). Unless extremely small time steps are taken, conventional analog Monte Carlo fails and goes unstable for this type of simulation again due to the close coupling of the radiation and matter energy density through the absorption and emission terms of the respective equations. To illustrate this problem, consider a two region geometry: one cold, the other hot. The hot region emits x-ray photons that are absorbed in the cold region causing its temperature to increase. As the cold region gets hotter, it in turn emits photons. This occurs nearly instantaneously, the rate being limited only by the speed of light and the distance separating the two regions. In a conventional Monte Carlo simulation, the emission from the recently heated region cannot occur until the time step *after* the absorption occurs. If temperatures change too much from one time step to the next, the simulation goes unstable. Reducing the time step can eliminate this stability problem, but at a cost of making the simulation too expensive. The alternate, innovative solution employed by IMC is to estimate the emission rate at the *future* matter temperature. The algorithm becomes *implicit* in temperature.

7.1. DERIVATION OF IMC

Demonstrating how to make the algorithm implicit is simplified by considering the grey, one dimensional formulation of the photon transport equation, where the energy dependence has been eliminated by integrating over all energy:

$$\frac{1}{c}\frac{\partial I}{\partial t}+\mu\frac{\partial I}{\partial x}+\sigma I=\frac{1}{2}c\sigma aT^4.$$

The Planckian emission term in both the transport equation and the material energy density equation,

$$\frac{\partial u_m}{\partial t}=\sigma\left(\int_{-1}^{1} I d\mu - acT^4\right),$$

shows the familiar T^4 dependence after integrating over energy. Defining, for convenience, the scaled source

$$B(T)=aT^4$$

and β

$$\beta = \frac{\partial B}{\partial u_m} = \frac{\partial B}{\partial T}\frac{\partial T}{\partial u_m} = \frac{4aT^3}{b}$$

and applying the chain rule to the matter energy balance equation results in the following expression for the scaled source:

$$\frac{\partial u_m}{\partial t} = \frac{\partial u_m}{\partial B}\frac{\partial B}{\partial t} = \frac{1}{\beta}\frac{\partial B}{\partial t} = \sigma\left(\int_{-1}^{1} I d\mu - cB(T)\right).$$

By integrating this equation over the time step $\Delta t = t^{n+1} - t^n$ and defining a time-centered value of *B* defined as

$$B^\gamma = \alpha B^{n+1} + (1-\alpha)B^n,$$

the scaled source equation becomes

$$B^\gamma - (1-\alpha)B^n - \alpha B^n = \alpha\bar{\beta}\bar{\sigma}\Delta t\left(\int_{-1}^{1} I^\lambda d\mu - cB^\gamma\right)$$

after eliminating the end-of-timestep value $B^{\mathrm{n+1}}$. Note that the superscripts λ and γ denote a time centering and the bars over β and σ denote their time average values.

Solving for the time-centered value of *B* and dropping the overbars yields

$$B^\gamma = \frac{\alpha\beta\sigma\Delta t}{1+\alpha\beta c\sigma\Delta t}\int_{-1}^{1} I^\lambda d\mu + \frac{1}{1+\alpha\beta c\sigma\Delta t}B^n.$$

Substituting this back into the transport equation and replacing the time centered intensity with its instantaneous value results in the implicit form of the transport equation,

$$\frac{1}{c}\frac{\partial I}{\partial t} + \mu\frac{\partial I}{\partial x} + \sigma I = \frac{1}{2}\sigma_s^{eff}\int_{-1}^{1} I d\mu + \frac{1}{2}c\sigma_a^{eff}B^n$$

where effective scattering and absorption cross sections are defined

$$\begin{aligned}\sigma_s^{eff} &= \frac{\alpha\beta c\sigma\Delta t}{1+\alpha\beta c\sigma\Delta t}\sigma \\ \sigma_a^{eff} &= \frac{1}{1+\alpha\beta c\sigma\Delta t}\sigma\end{aligned}.$$

Replacing the total cross section with the effective absorption cross section similarly modifies the material energy density equation,

$$\frac{\partial u_m}{\partial t} = \sigma_a^{eff}\left(\int_{-1}^{1} I d\mu - cB^n\right).$$

Note how the effective scattering and absorption cross sections behave with varying α and Δt. As α goes to zero, i.e., the method becomes fully explicit, the effective scattering cross section vanishes and the effective absorption cross section goes to the total cross section. IMC reverts to conventional Monte Carlo. The same is true as the length of the time step shrinks to zero. As the length of the time step goes to infinity, the effective absorption cross section vanishes, the effective scattering cross section goes to the total cross section, and implicit transport equation reverts to the transport equation under conditions of radiative equilibrium:

$$\frac{1}{c}\frac{\partial I}{\partial t} + \mu\frac{\partial I}{\partial x} + \sigma I = \frac{1}{2}\sigma\int_{-1}^{1} I d\mu$$

In the IMC simulation procedure, a fraction of the absorption in the current step and emission in the next step has been replaced by an effective scattering in the current time step. The fraction is dictated by the effective scattering cross section which is determined by a number of terms including two that can be controlled by the user: the fraction of “implicitness”, α, and the time step, Δt.

7.2. PARALLEL SOLUTION STRATEGY

Currently, KULL employs two different schemes for parallelizing IMC. For geometries small enough to fit on a single processor, a technique known as complete geometry replication is used. Instead of decomposing the problem over space, as is done for hydrodynamics and discrete ordinates, parallelization is achieved by decomposing over particles. The geometry is replicated and the IMC step is performed independently on each processor. Care must be taken to ensure that the random number sequences on each processor are indeed independent. After each processor completes its simulation the results are summed together to obtain a result with statistical accuracy consistent with the total number of particles run.

For ASCI-scale problems this technique is infeasible because the geometry is too large to fit on each processor. Instead, spatial domain decomposition is used where each spatial subdomain is assigned to a single processor. As particles leave a subdomain during the IMC simulation, they are batched into messages and sent to the processors containing the neighboring subdomains. There the particles continue their transport.

We hope to explore a hybrid of these two techniques in the future. The geometry could be split into subdomains just small enough to fit on individual processors. The subdomains could then be replicated over processors. These processors would solve independent Monte Carlo simulations on the subdomains exploiting decomposition over particles. The number of processors assigned to each subdomain would be controlled by the amount of time required to complete each subdomain’s work. Varying the number of processors per subdomain would allow the work to be balanced across processors.

7.3. PARALLEL PERFORMANCE

The geometry replication option shows excellent parallel efficiencies as expected. Figure 13 shows the relative run time for a problem where the number of particles in the simulation increases with the number of processors. Performance improves with more particles per processor because the fraction of time spent in communication decreases as the time spent doing computation increases.

The domain decomposed IMC exhibits poorer parallel performance for the sample test problem of a slab of material with a temperature source at one end. Figure 14 shows the execution times decreasing as a function of the number of processors until 256 processors. In this plot the total number of particles for a particular curve is held constant. As the number of processors increases each spatial domain becomes smaller. The departure from optimal performance is due to two factors. First, the communication overhead for the domain decomposition cases is significantly greater than for geometry replication because communication between nodes is necessary throughout each time step as particles move from domain to domain. For the geometry replication case, communication occurs only at the beginning and end of the time step. Second and probably the most important contribution to the degradation in performance, the workload between processors is out of balance for this problem. Because particles are concentrated at the hot end of the slab and subdomains are distributed uniformly along the slab, processors assigned to subdomains closer to the hot end do much more work than processors assigned to regions at the other end.

CONCLUSIONS

KULL’s three dimensional, time dependent hydrodynamics and radiative transfer methods have been designed to work well on unstructured meshes of arbitrarily connected zones of polyhedra. The hydrodynamics is a flexible ALE method which is sufficiently robust to handle the turbulent flows encountered in ICF experiments. KULL can simulate the transfer of radiation resulting from the emission of x-rays by hot matter through any one of three techniques: diffusion for quick, less accurate simulations and discrete ordinates and Implicit Monte Carlo when full-fledged, angular dependent transport solutions are required, albeit at a computational cost. Although not presented here, integrated radiation hydrodynamics capability is just becoming available and will represent a valuable computational tool to the ICF community and the Stockpile Stewardship Program.

ACKNOWLEDGEMENTS

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

REFERENCES

1. Gropp, W., Lusk, E., Skjellum, A., Using MPI: Portable Parallel Programming with the Message Passing Interface, MIT Press (1994).

2. OpenMP Architecture Review Board, OpenMP FORTRAN Application Program Interface (1997).

3 . G.C. Pomraning, *The Equations of Radiation Hydrodynamics*, Pergamon, Oxford (1973).

4. T.S. Palmer, “A Point-Centered Diffusion Differencing for Unstructured Meshes in 3-D,” *Proc. Int. Conf. Mathematics and Computation, Reactor Physics and Environmental Analysis*, Portland, Oregon, April 30-May 4, 1995, Vol. 2, 897 (1995).

5. D.E. Burton, “Conservation of Energy, Momentum, and Angular Momentum in Lagrangian Staggered-Grid Hydrodynamics,” Lawrence Livermore National Laboratory Report UCRL-JC-105926 (1991).

6. S. Balay, W. D. Gropp. L. C. McInnes, and B. F. Smith, “Efficient Management of Parallelism” in *Object-Oriented Numerical Software Libraries*, Birkhauser Press 163 (1997).

7. E. Chow, A. J. Cleary, and R. D. Falgout, “Design of the HYPRE preconditioner library,” *Proc. SIAM Workshop on Object Oriented Methods for Inter-operable Scientific and Engineering Computing*, Yorktown Heights, New York, October 21-23, 1998 (also UCRL-JC-132025).

8. Y. Saad, *Interactive Methods for Sparse Linear Systems*, PWS Publishing Company, Boston (1996).

9. A. J. Cleary, R. D. Falgout, V.E. Henson, and J. E. Jones, “Coarse-grid selection for parallel algebraic multigrid,” *Proc. Fifth Int. Symposium on Solving Irregularly Structured Problems in Parallel*, Berkeley, CA, August 9 - 11, 1998 (also UCRL-JC-130893).

10. M.L. Adams, “Subcell Balance Methods for Radiative Transfer on Arbitrary Grids, *Transport Theory Statist. Phys*., **26**, 385, (1999).

11. E.W. Larsen, “A Grey Transport Acceleration Method for Time-Dependent Radiative Transfer Problems,” *J. Comput. Phys.*, **78**, 459 (1988).

12. G.L Ramone, M.L. Adams, P.F. Nowak, “Transport-synthetic Acceleration methods for Transport Iterations,” *Nucl. Sci. Eng.*, **125**, 257, 1997.

13. P.F. Nowak and M.K. Nemanic, “Radiation Transport Calculations on Unstructured Grids Using a Spatially Decomposed and Threaded Algorithm,” *Proc. Int. Conf. Mathematics and Computation, Reactor Physics and Environmental Analysis in Nuclear Applications*, Madrid, Spain, September 27-30, 1999, Vol. 1, p. 379 (1999).

14. J.A. Fleck, Jr. and J.D. Cummings, “An Implicit Monte Carlo Scheme for Calculating Time and Frequency Dependent Nonlinear Radiation Transport,” *J. Comput. Phys.*, **8**, 313 (1971).

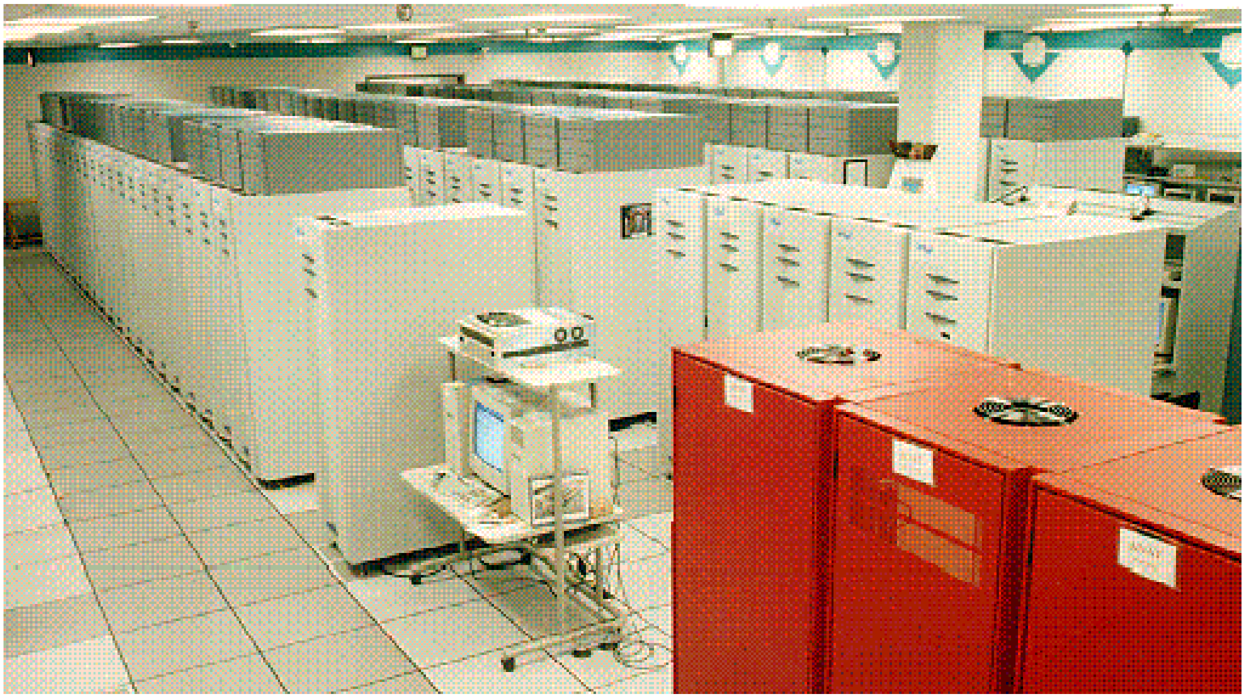

Figure 1. ASCI Red at Sandia National Laboratories.

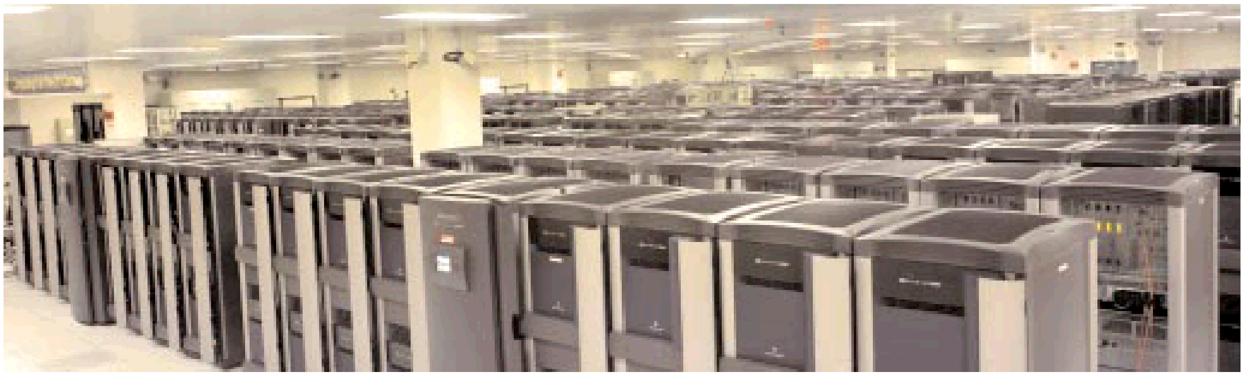

Figure 2. ASCI Blue Mountain at Los Alamos National Laboratory.

ASCI Option Blue – Pacific
Lawrence Livermore National Laboratory

Figure 3. ASCI Blue Pacific at Lawrence Livermore National Laboratory.

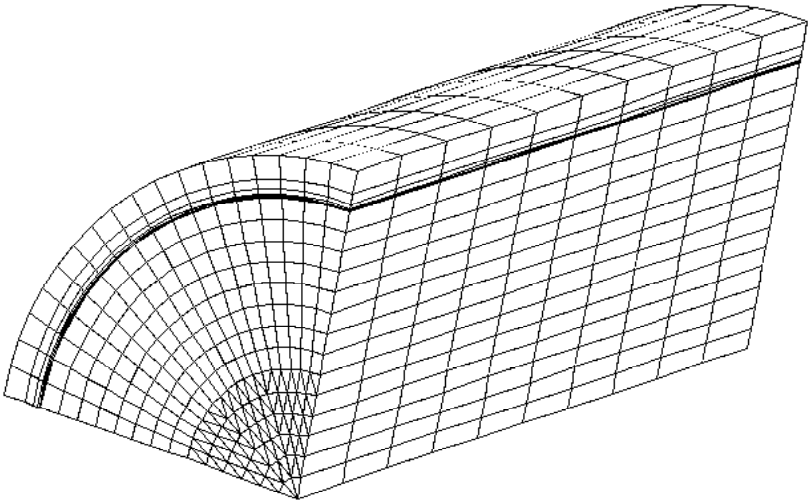

Figure 4. KULL mesh showing arbitrary connectivity near center.

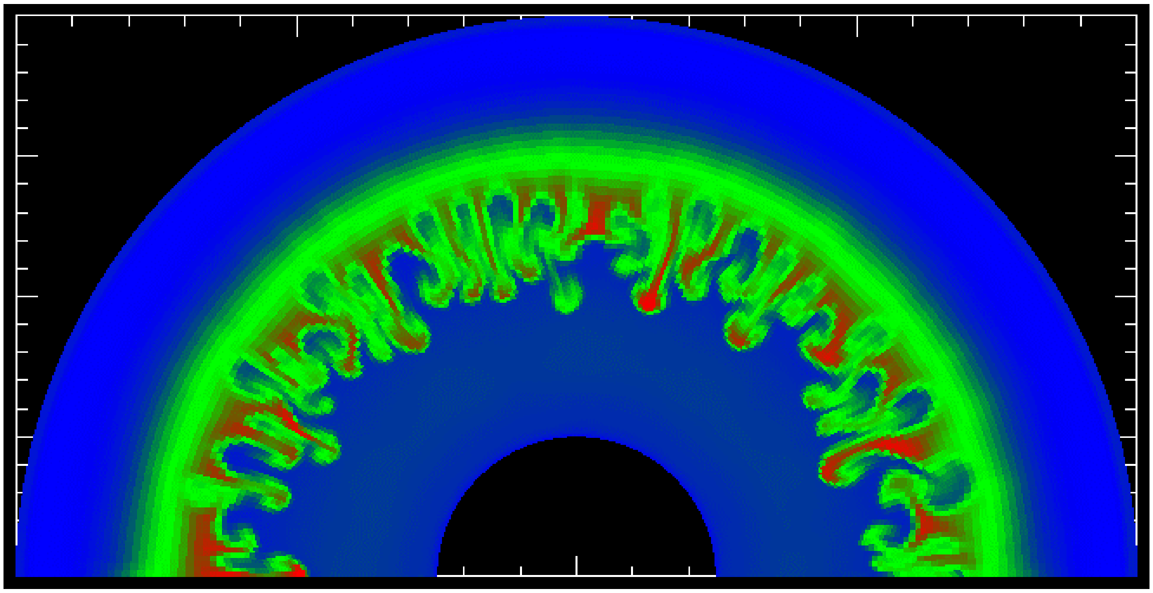

Figure 5 A KULL ALE simulation of Richtmyer-Meshkov instability in imploding ICF system.

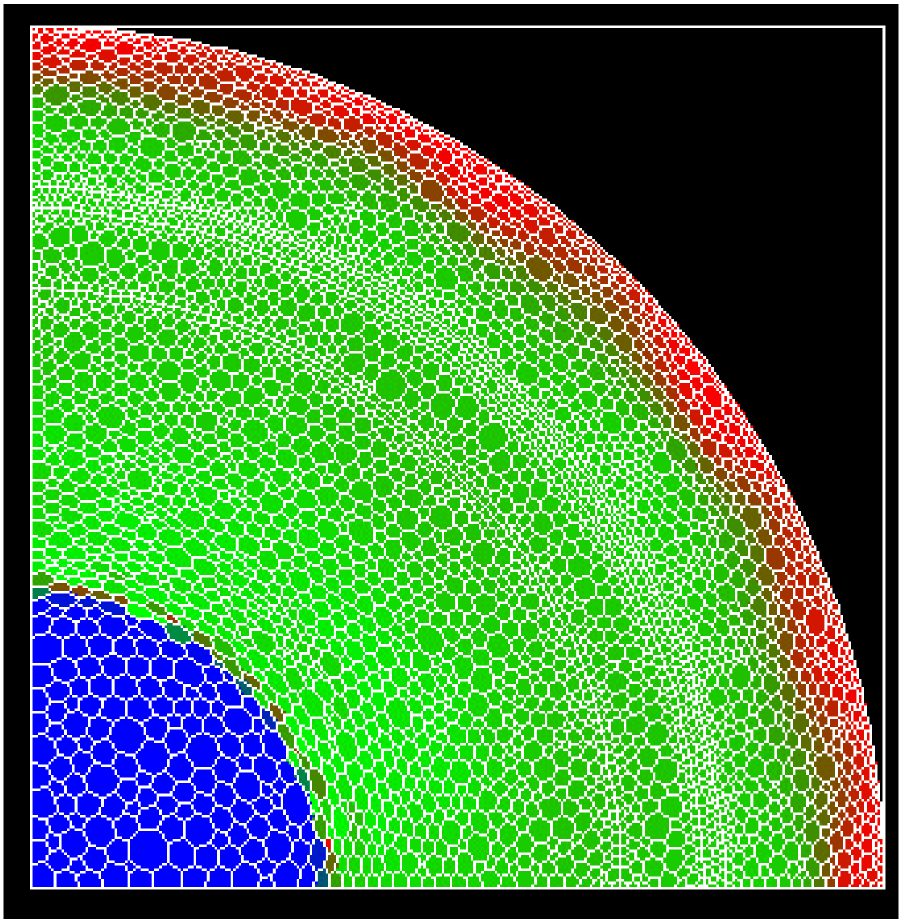

Figure 6. Imploding ICF system with unstructured Voronoi mesh.

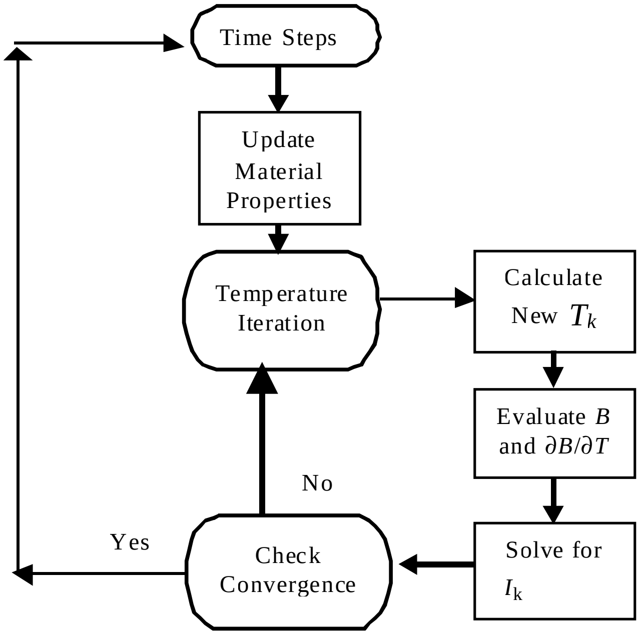

Figure 7. Flowchart of the radiative transfer iteration.

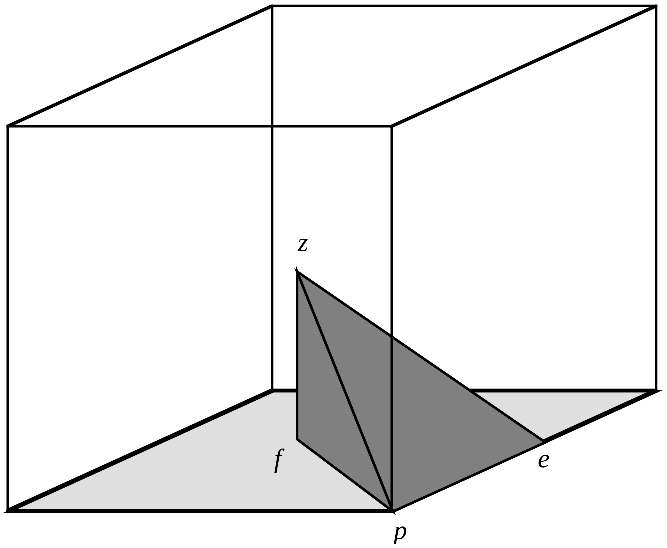

Figure 8. A wedge in a cubic zone.

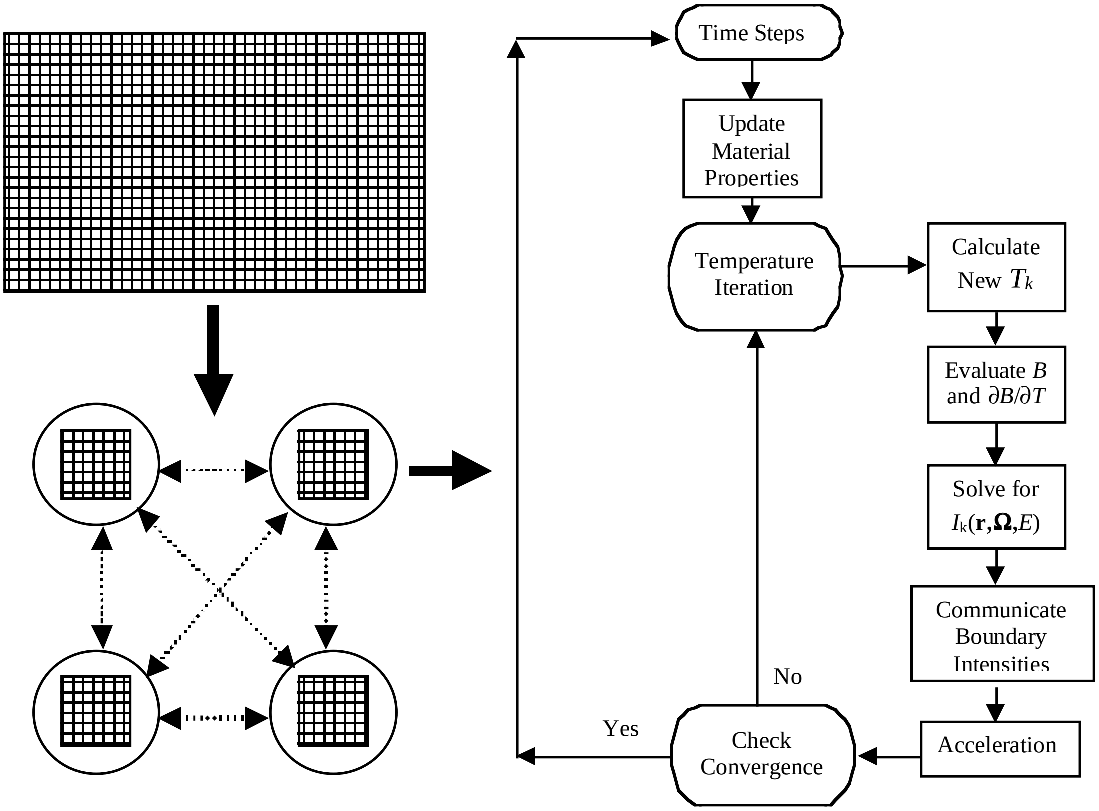
Time Steps
Update Material Properties
Temperature Iteration
Calculate New T_k
Evaluate B and $\partial B/\partial T$
Solve for $I_k(\mathbf{r},\mathbf{\Omega},E)$
Communicate Boundary Intensities
Acceleration
Check Convergence
No
Yes

Figure 9. Flowchart of the radiation transport iteration on a spatial sub-domain.

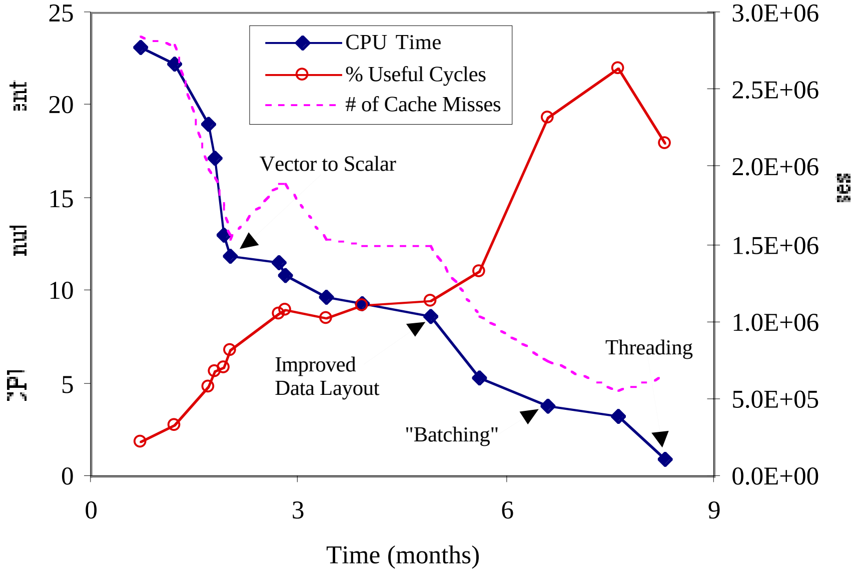

Figure 10 Improvements in discrete ordinate transport solution serial run time and cache efficiency.

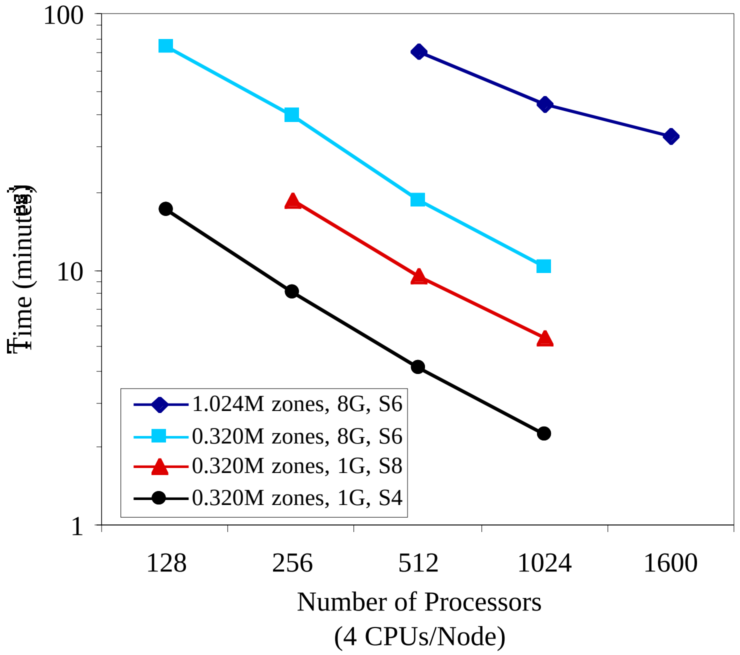

Figure 11. Scaling results for a constant problem size using MPI and threads for discrete ordinate transport.

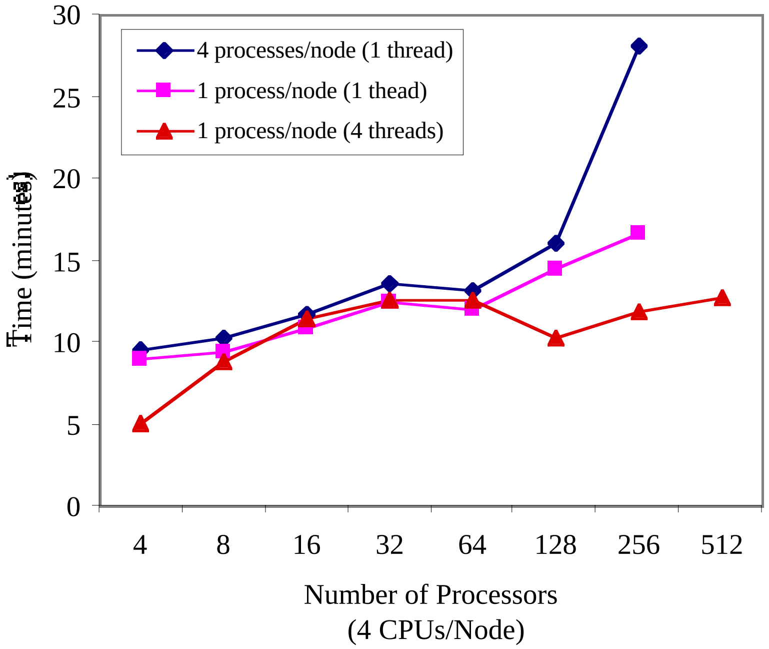

Figure 12. Scaling results for constant work per processor for discrete ordinate transport.

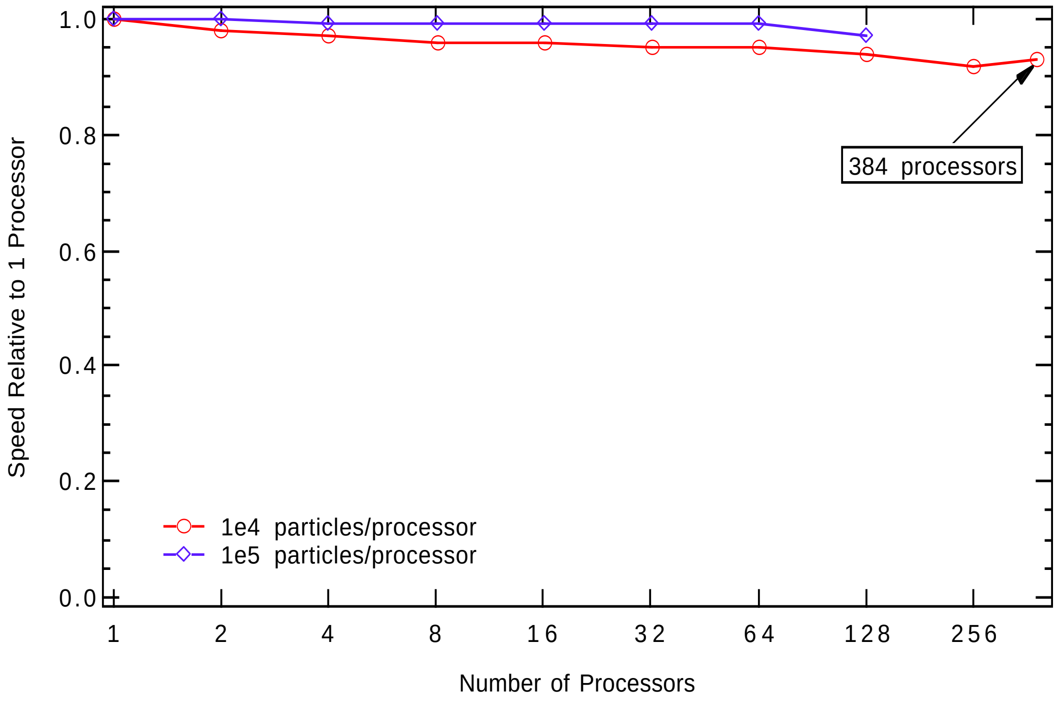

Figure 13. Scaling results for constant work per processor with geometry replication IMC.

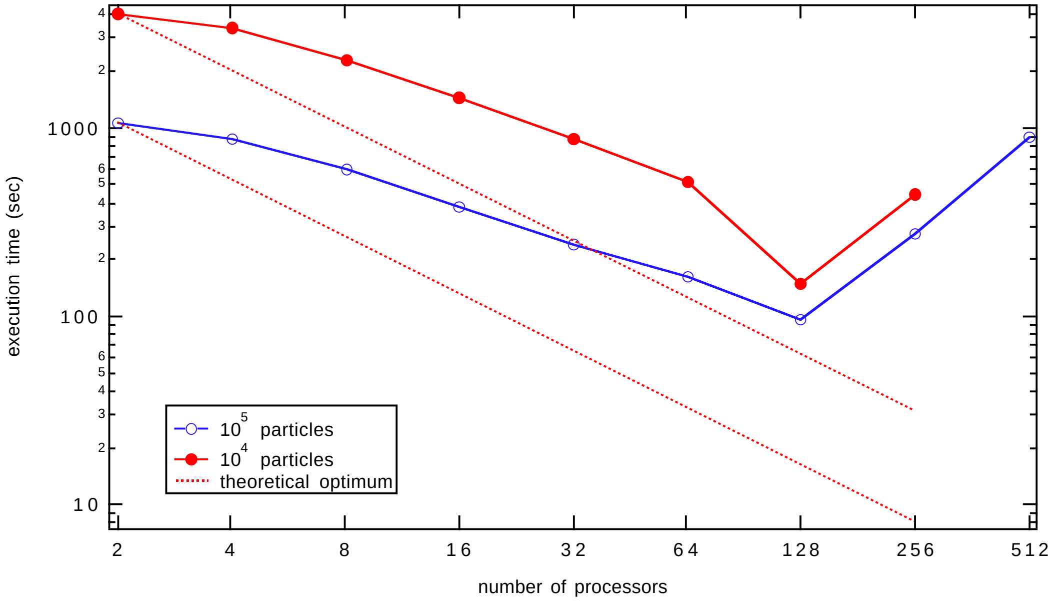

Figure 14. Scaling results for constant total work with domain decomposed IMC.